Contents

Introduction to Co-Processors

- ***A Co-processor*** is specially designed circuit on microprocessor chip which can perform the same task very quickly which the microprocessor perform.
- ***The main advantage of co-processor is to reduce the workload of the main processor.***
- Co-processor share the same memory, I/O system, bus, control and clock generator.
- Co-processor handles specialized task like mathematical calculation, graphical display on screen etc.

8087 Coprocessor

- ***It was the first arithmetic/math coprocessor designed by Intel to pair with 8086/8088 resulting in easier and faster calculation.***
- The **purpose of 8087** was to speed up the computation involving floating point calculation. It can work with integer, decimal, real, floating point numbers.
- It is capable of doing complex arithmetic and trigonometric calculations.
- It is basically a secondary processor which has its own architecture and instruction set.
- It is available in **40 pin IC**. it is an **80 bit processor**.

Features of 8087 Coprocessor

- This increases the **overall speed** and **system performance** of the entire system.
- Addition, Subtraction, Multiplication and Division of the simple numbers is not the coprocessor's job.
- It does all the calculations involving floating point numbers like **Scientific calculation** and **algebraic function**.
- By having a coprocessor, which perform all the calculations, it can **free up a lot of CPU time**.
- This would allow the CPU to focus all of it resources on the other function it has to perform.

Instructions of 8087 Coprocessor

- This coprocessor introduced about **60 new instructions** available to the programmer.
- All the mnemonics begin with **"F"** to differentiate them from the standard 8086 instructions.

  ADD ----> FADD

  MUL ----> FMUL

- If instruction is an ESCape (coprocessor) instruction, the coprocessor executes it, if not the microprocessor executes.
- Instructions of 8087 are treated as NOP instructions by 8086.

Why Arithmetic CoProcessor?

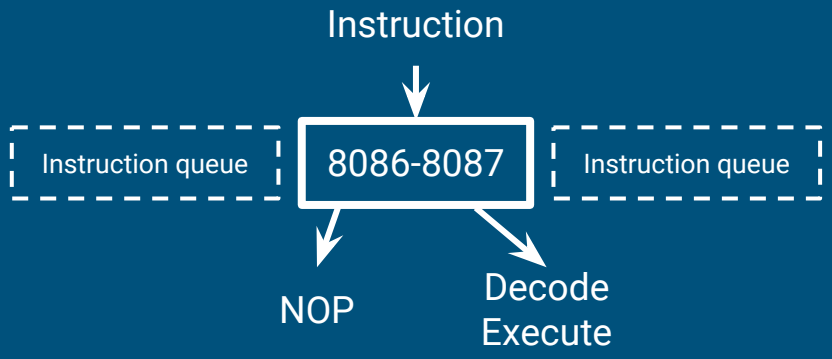
- Floating point operations are complex math operation that required large registers, complex circuits and area on the chip.
- A general processor avoids this much burden and delegates such operations to a coprocessor designed specifically for this purpose. Example: 8087 coprocessor for 8086 microprocessor.
- 8087 can perform a mathematical computation 100 times faster than the 8086 and 8088.

Interfacing of 8087 with 8086

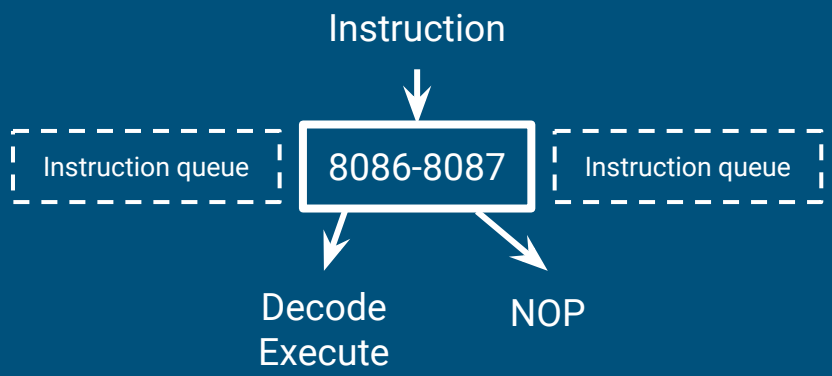
Interfacing of 8087 with 8086

- Homogeneous program (8087 + 8086 instructions) is written in memory.
- Each processor decodes all instructions in the fetched instruction byte stream but executes only its own instruction.

If an 8087 instruction comes in:



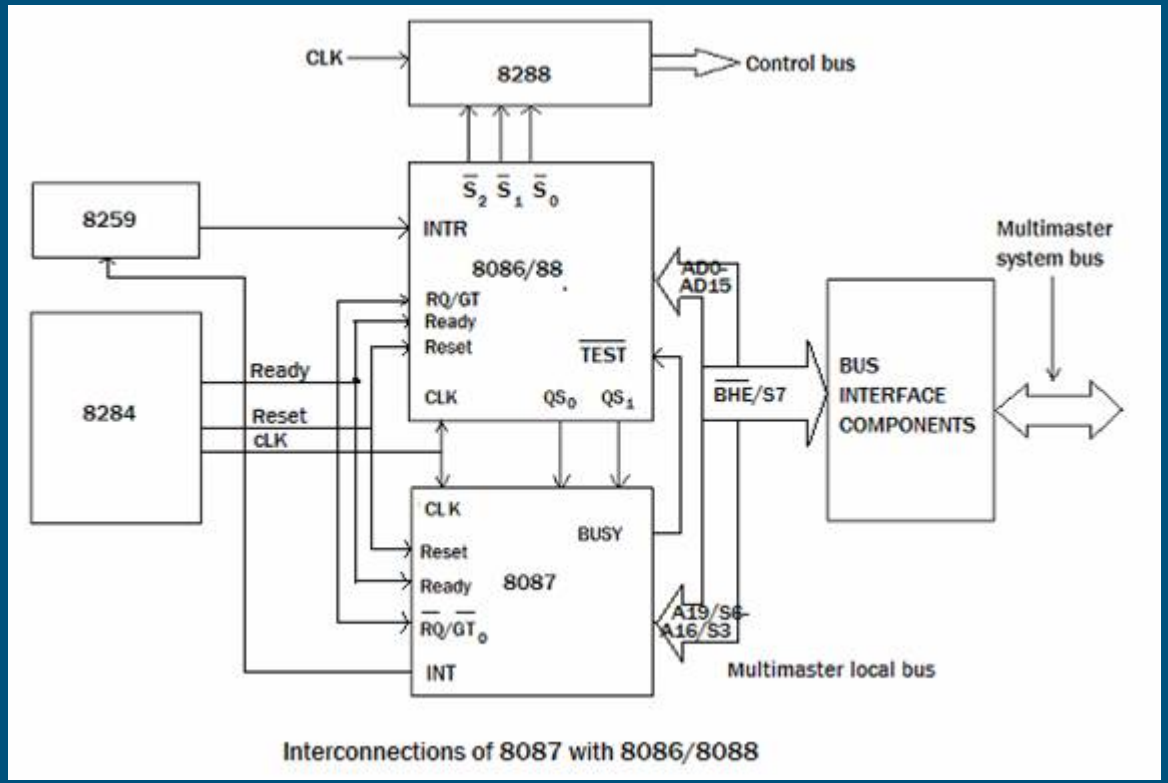
If an 8086 instruction comes in:



Interfacing of 8087 with 8086

8259
Programmable
Interrupt Controller

8284
Clock Generator



Interconnections of 8087 with 8086/8088

8288
Bus Controller

8086/8088
Host Microprocessor

8087
Co processor

Interfacing of 8087 with 8086

- 8086 is called the host processor. 8087 is connected as a co-processor to 8086.
- 8087 uses I/O system, bus system, memory, clock inputs of 8086.

➔ 8086 operates in two modes. (Max and Min mode)

- ◆ MN/MX' pin (0⟶ Max mode)

- **8084:** It provides the CLK, READY and RESET signals to both host and co processor.
- **8288:** It generates control signals from S0, S1, and S2 signals.
- **8259:** It accepts interrupts from 8087 and sends it to 8086/8088.

Interfacing of 8087 with 8086 (1/3)

1. As a coprocessor (8087) is connected to 8086, 8086 operates in maximum mode. Thus the **MN/MX' is grounded**.
2. This interface is also called as **coprocessor configuration**.
3. We write a **homogeneous program** which contains both 8086 as well as 8087 instructions.
4. **Only 8086 can fetch instructions** but these instructions also enter 8087. 8087 treats 8086 instructions as NOP.
5. **ESC is used as a prefix for 8087 instructions**. When as instruction with ESC prefix is encountered, 8087 is activated. The ESC instruction is decoded by both 8086 and 8087.

Interfacing of 8087 with 8086 (2/3)

6. If the 8087 instruction **has only an opcode** then 8087 will start execution and 8086 will immediately move its next instruction.
7. But if the instruction requires a memory operand, then 8086 will have to fetch the first word of the operand as 8087 cannot calculate the physical address. This word will be captured by 8087. Now the remaining words can be fetched by 8087 by simply incrementing the address of the first word. Thus **8087 need help from 8086**.
8. Once 8087 gets its operand, it begins processing by making the BUSY output high. This BUSY output is connected to the TEST input of the microprocessor. Now 8087 execute its instruction and 8086 moves ahead with its next instruction. Hence **multiprocessing takes place.**

Interfacing of 8087 with 8086 (3/3)

9. The QS0 and QS1 lines are used by 8087 to **monitor the queue of 8086**.
10. 8087 uses the **shared system bus** to perform the data transfer with the memory.
11. The RQ'/GT0' of 8087 is connected to RQ'/GT0' of the microprocessor to **maintain read/write bus cycle**.
12. During the execution **if an exception occurs**, 8087 interrupts microprocessor using the INT output pin through the PIC 8259.

Thank You